NJIT
New Jersey's Science & Technology University
THE EDGE IN KNOWLEDGE

CS 280 Programming Language Concepts

Hello, World

NJIT New Jersey's Science & Technology University COLLEGE OF COMPUTING SCIENCES

Python

```
import sys

if __name__ == "__main__":
        # command line arguments are in sys.argv
        print("Hello, world!")
```

Java

```
class HiWorld {
        public static void main(String argv[]) {
                System.out.println("Hello, world!");
        }
}
```

C

```
#include <stdio.h>

int
main(int argc, char *argv[])
{
        printf("Hello, world!!!\n");
        return 0;
}
```

C++

```
//====================================================================
// Name        : HelloWorld.cpp
// Description : Hello World in C++
//====================================================================
// stuff that begins with // is a comment

/* this is also a comment */

#include <iostream>
using namespace std;

int main(int argc, char *argv[]) {
        cout << "Hello there, C++ programmers!" << endl;
        return 0;
}
```

main

- Every C++ program must have one (and only one) function named main
- The main function is where the program starts
- When main is finished, it returns and the program is done

functions

- To define a function you must specify
  - function name
  - type for the value that that the function returns when it is done
  - arguments (name and type) that are passed in to the function
  - the body of code that is executed when the function is called

```
// the code below says
// main is the name of the function
// the type that it returns is int
// main takes two arguments named argc and argv
// the two statements inside of the { } are run when main gets called

int main(int argc, char *argv[]) {
        cout << "Hello there, C++ programmers!" << endl;
        return 0;
}
```

Calling functions

- To call a function that you or someone else has defined, you need to know its name, return type, and arguments
- You can tell the compiler about a function that you want to call

```
extern int main(int argc, char *argv[]);
```

classes

- Java programmers: notice that there is no "class" keyword in this program
- In C++, everything does NOT have to be in a class
- We don't need a class for this simple example. We will just write a main function.
- Functions do NOT have to be methods defined inside of a class.

Preprocessor

- #include is a *preprocessor directive*
- It tells the compiler to include the contents of a file into the body of the code before compiling
- A filename inside of < and > means that the file is in the standard place for the standard files that come with the compiler
- You can make your own include files and #include them, enclosing filenames in " and "
- By convention, these included files are called "header files" and usually have names ending in .h

Header or Include Files

- C and C++ rely on including files in the preprocessor phase of the compile to make sure that all programs have common definitions of things: variables, constants and different types
- Header files are often associated with library implementations
- C++ comes with a lot of standard libraries and header files
- For the most part, it is a bad idea to #include a .cpp file

using namespace

- C++ can put functions and definitions into a separate, labeled *namespace*
  - Java programmers: this is a little like packages
- To access something in a namespace, you need to preface the item you are accessing with the name of its namespace and ::
- Items in the C++ standard library are in a namespace named “std”, so cout is actually std::cout
- Saying “using namespace std;” tells the compiler that you want to have everything in the std namespace visible in your program
- You could also indicate that you’re just using one symbol: “using std::cout;”

iostream

- iostream is a C++ standard library that provides a definition and implementation for input streams (istream) and output streams (ostream)
- At runtime every program has:
  - “standard input” or “standard in” (in Java, System.in)
  - “standard output” or “standard out” (in Java, System.out)
  - “standard error” or “standard err” (in Java, System.err)
- To read what a user types in, read the standard in
- To write something for the user to see, write to standard out

iostream

- The iostream library provides a type-safe way to access standard in and standard out.
- cin is an input stream connected to standard in
- cout is an output stream connected to standard out
- cerr is an output stream connected to standard err

- iostream uses the << operator to write to a stream
- iostream uses the >> operator to read from a stream
- There are also methods defined for various operations on streams

```
// the line below performs the << operation on cout
// stream << something
//     causes the something to be written to the stream
//     in this case, the something is a string of characters
// << endl
//     causes and end of line to be written to the stream
//     if you leave off endl you will not skip to the next line
//     writing "\n" or '\n' does the same thing

       cout << "Hello there, C++ programmers!" << endl;
```

NJIT New Jersey's Science & Technology University   COLLEGE OF COMPUTING SCIENCES

operator << and >>

- The implementors of iostream decided to use the C/C++ logical bit shift operators << and >> to mean "shift information into and out of the stream"
- The << operator is defined to "shift information out to the stream
- The >> operator is defined to "shift information in from the stream"
  - So this code reads in an integer:

```
int x;
cin >> x;
```

- Reusing operators in this way is called "operator overloading". It's a feature of C++

NJIT New Jersey's Science & Technology University   COLLEGE OF COMPUTING SCIENCES

Errors with streams

- Errors might happen:
  - "End of file" is reached
  - The input on the stream cannot be converted to the proper type
- You need to check for errors!

Stream methods for errors

- good() is true if there are no errors
- eof() is true if the end of file was reached
- fail() is true if there was a logical error or a read/write error on the stream
- bad() is true if there is a read/write error on the stream

Streams without error checking

```
#include <iostream>
using namespace std;

// run until user enters a -1
int
main(int argc, char *argv[])
{
        int     ival = 0;

        for(;;) {
                cin >> ival;

                cout << "You entered " << ival << endl;

                if( ival == -1 )
                        break;
        }

        cout << "Bye!" << endl;
        return 0;
}
```

Streams with error checking

```
// run until user enters a -1
int
main(int argc, char *argv[])
{
        int     ival = 0;

        for(;;) {
                cin >> ival;

                if( cin.eof() )
                        break;

                if( cin.fail() == false )
                        cout << "You entered " << ival << endl;

                if( ival == -1 || cin.fail() )
                        break;
        }

        cout << "Bye!" << endl;
        return 0;
}
```

Some istream methods

- get - read a single character
- get – read a sequence of characters into an array of characters
- getline - read a line (a sequence of characters terminated by a newline) into an array of characters
- Also a getline function to read from a stream into a "string"

Example

```
// copy standard input to standard output,
// one character at a time

#include <iostream>
#include <fstream>
using namespace std;

int main(int argc, char *argv[])
{
        int     ch;

        while( (ch = cin.get()) != EOF ) {
                cout.put(ch);
        }

        return 0;
}
```

Read input a line at a time

```
#include <iostream>
#include <string>
using namespace std;

int main()
{
        string inLine;

        for(;;) {
                getline(cin, inLine);

                if( !cin.good() )
                        break;

                cout << "You typed:" << inLine << endl;
        }

        return 0;
}
```

NJIT New Jersey's Science & Technology University

COLLEGE OF COMPUTING SCIENCES

NJIT

THE EDGE IN KNOWLEDGE